An Adaptive-reliability Cyber-physical Transport Protocol for Spatio-temporal Data

Hossein Ahmadi and Tarek Abdelzaher
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, IL
{*hahmadi2, zaher*}*@uiuc.edu*

***Abstract*—In this paper, we propose a new transport protocol for data collection in sensor networks that monitor physical phenomena. In a network with variable channel condition, this protocol adapts transmission reliability based on the importance of transmitted spatio-temporal data to the reconstruction of the phenomenon. Data whose omission generates a higher estimation error are transmitted more reliably. The protocol aggregates data from nodes to the base station providing a constant expected estimation error while significantly reducing the energy consumption and communication overhead compared with other approaches to reliable communication. Moreover, our protocol is easy to implement on current motes. To the best of our knowledge, this is the first transport layer protocol, that incorporates predictive models, to be implemented on motes. We perform extensive experiments on MicaZ motes using LiteOS to compare the performance of our protocol against reliable transport protocols. Our experimental results show that our protocol can keep the estimation error at very small levels while saving orders of magnitude in consumed energy.**

I. Introduction

This paper presents a new transport layer for data collection in sensor networks that defines the degree of (partial) reliability of communication in terms of a bound on loss-induced error in estimating the physical phenomenon reported. The protocol is based on the observation that the importance of reliable transmission depends on the utility of transmitted data. Since natural phenomena obey physical models, it may be possible to estimate future measurements rather than carry them across the network. Hence, data are valuable only to the extent to which they cannot be estimated accurately from the model. In other words, data are valuable only to the extent to which they reduce model estimation error. Our protocol bounds the overall inaccuracy of sensing the physical environment in an unreliable network. The desired error bound can be chosen by the user. A zero-error protocol corresponds to fully reliable transmission.

Optimizing transport to the unique characteristics of sensor networks has received a lot of attention in past literature [1]–[8]. There are several reliable sensor network protocols that employ traditional end-to-end packet delivery guarantees [1]–[4]. There have also been prior efforts to introduce new reliability semantics for data transport in monitoring applications [6]–[8]. These prior approaches have two major weaknesses; namely, (i) they have only limited knowledge of the real importance of a data packet from the application perspective, and (ii) they usually rely on feedback from the receiver to adapt the amount of packets delivered. The second issue is a consequence of the first, and imposes communication overhead on the network. In contrast, our protocol provides a constant expected estimation error while significantly reducing the energy consumption and communication overhead.

There has been a lot of work in sensor networks on adaptive sampling of a physical environment to provide an accurate estimation with the minimum number of samples [9], [10]. Also, several works address data reduction using prediction models [11], [12]. In particular, these schemes send information only if it can not be obtained from physical models accurately. The major difference between data reduction techniques (e.g. [11], [12]) and ours is that these protocols assume reliable communication and hence the decision as to which packets to suppress is entirely up to the protocol. In contrast, we do not assume a reliable medium and only influence packet delivery probabilities where channel losses ultimately determine the delivery of packets. This stochastic quality introduced by an unreliable medium has not been addressed in any data reduction technique. Furthermore, unlike those works, we are the first to implement ours on the motes. Our protocol is simple yet general enough to be used with any monitoring application.

In this paper, we develop a new transport protocol for wireless sensor networks that uses a specified bound on estimation error as a metric to determine the amount of information to be reliably delivered. It is called cyber-physical because it takes into account the physical importance of packets containing spatio-temporal measurement data. The main idea is to preferentially retransmit and hence increase the delivery probability of packets that significantly improve reconstruction of the estimated phenomenon. This can be done without global information while achieving the global bounded-error requirement. Our protocol reaches the desired estimation error with the least energy consumption without feedback from the basestation or synchronization among nodes. Avoidance of feedback is important because it reduces communication overhead and allows the protocol to be applied to very dynamic phenomena, where the past is not a good indication of the future and feedback schemes

Table I
NETWORK LAYER API

Method
function *send*(destination_id, datagram)
callback *receive*(source_id, datagram)

Table II
TRANSPORT LAYER API

Method
listen(port_number, expected_error, model_parameters)
connect(port_number, destination_id)
send(port_number, destination_id, payload)
send_data(port_number, destination_id, spatio_temporal_data)
receive(port_number)
close(port_number)

can not easily stabilize.

We implement our protocol in LiteOS on MicaZ motes and rigorously evaluate its performance through testbed experiments. We experimentally demonstrate that our protocol indeed provides the requested reliability. We also compare our protocol against packet-level reliable (RMST [2]), feedback-based (ESRT [6]), and unreliable transport protocols and show that we save a great amount of energy at the expense of producing a small error in the output.

The rest of this paper is organized as follows: In Section II, we present our system model and discuss the application and network layer interface. Section III describes the new transport protocol designed based on our reliability semantics. In Section IV, we discuss how the protocol is implemented on a MicaZ testbed running LiteOS. Then, we evaluate our protocol and compare it against other reliable and unreliable transport protocols in Section V. We summarize the related work in Section VI and finally conclude the paper in Section VII.

II. SYSTEM MODEL

Our transport protocol is designed to run on top of a network consisting of a set of sensors and a base station deployed to monitor a physical phenomenon. We assume that sensors perform readings and send the measurements to the base station for processing. The purpose of the transport protocol is to deliver monitoring information reliably to the base station. Since several WSN routing protocols are tree based [13], we present our protocol assuming that packets can be routed upward and downward the tree rooted at the base station. However, our protocol can work with any general routing protocol by maintaining tree information in transport layer. The routing layer invokes the transport at intermediate hops as well as the destination. Table I summarizes the API between our transport layer and the network layer.

Many monitoring sensor networks are designed to reconstruct a physical phenomenon at the base station. Every observation from any particular sensor at a specific time is represented as a sampling point in space and time. In the transport layer, we adapt transmission reliability based on the amount of estimation error contributed if the wireless link drops update messages. We use the expected value of estimation error as the reliability metric. In other words, a transport protocol is reliable if it can guarantee that the expected error is bounded by a desired constant at any time.

Our protocol is connection-oriented where each connection involves a subset of participating nodes in the network and supports data collection from these nodes to the same base station. Like other transport protocols, our protocol provides *listen* and *connect* functions to initiate the connection as depicted in Table II. Listen is called at the basestation while connect is called at those sensors participating in monitoring on the given port. Port numbers represent well-known ports and are used to distinguish between different applications running on the network. When initializing a connection, the base node needs to provide expected_error, the value of acceptable estimation error that the protocol should guarantee. On top of that, the application can optionally provide *model_parameters* to express the physical model of the phenomenon. Our protocol supports linear auto-regressive moving average (ARMA) models. If not provided with a model, the transport protocol assumes slow changing data. Hence, an estimated value equals the latest observed value (i.e. $x(t_{i+1}) = x(t_i)$).

The transport layer provides a specialized send function for sensor data, called *send_data*, to send spatio-temporal measurements to the basestation. The spatio_temporal_data payload has a specific format: It contains an array of measurements each preceded by a header involving a timestamp (of first sample), a data type ID, and sampling rate. For example, payload can contain all the values read from the temperature sensor of the node within some interval. When using this function, a slightly different version of the measurements may be returned by the *receive* function at the basestation. However, the total difference is guaranteed not to exceed the value of the acceptable estimation error specified in the listen call. In order to send other arbitrary data (e.g., send control information), we also provide a regular *send* function. Its payload is transmitted 100% reliably. Finally, the *close* function is used by the base station to close the connection.

III. PROTOCOL DESIGN

The main challenge in designing an energy efficient cyber-physical transport protocol is to quantify the importance of a packet and coordinate the transmissions in such a way that the global estimation error is bounded. This should be done with almost no communication overhead since conserving energy is our main objective. Moreover, our design should be able to handle general packets with 100% reliability requirements. We choose a probabilistic transmission ap-

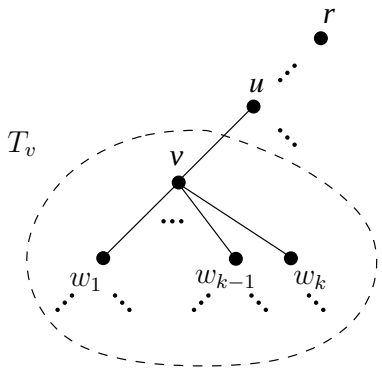


Figure 1. Routing tree and notations.

proach using local decisions to guarantee the resulting error in expected value sense.

In order to describe how our protocol efficiently provides reliability, we first need to present the notation used in the rest of this paper. Given the routing tree rooted at base station, r, let T_v be the sensor network subtree rooted at node v (Figure 1). Let $M_v(t) = \{m_w(t)|w \in T_v\}$ be the measured phenomenon in the region covered by T_v at time t where $m_w(t)$ is the actual reading of the sensor at node w. In our protocol, every node v has an estimation of space-time data collected in $T_z \subseteq T_v$. We define $\bar{M}_z^v(t) = \{\bar{m}_w^v(t)|w \in T_z\}$, to be the estimated values of phenomenon in the region T_z and at time t which is present to node v. $\bar{m}_w^v(t)$ is the value that node v locally thinks that w has measured. This value can come from either actual reading of node w or the result of a physical model estimation. Assuming that $c_1, \ldots, c_l$ denotes the coefficients of auto-regressive moving average physical model provided by the application, $\bar{m}_w^v(t_i) = c_1\bar{m}_w^v(t_{i-1}) + \cdots + c_l\bar{m}_w^v(t_{i-l})$ estimates future data from w based on the previous observations. $E_v(t) = \sum_{w \in T_v} (m_w - \bar{m}_w^v)^2$ is the estimation error at node v and defined as the square of error between $M_v(t)$ and $\bar{M}_v^v(t)$.

For any given value of ϵ, our protocol guarantees that the expected estimation error at root, $E[E_r(t)]$, does not exceed $n\epsilon$ at any time instance where n is the number of connected nodes. We first give an overview of the protocol explaining connection management and general-purpose API. Next, we present our approach for spatio-temporal data transport and we analytically prove the correctness of our approach in achieving the desired expected estimation error. Finally, we provide details of the protocol in case of variable link quality.

A. Connection Management and General API

Each node keeps track of established connections by maintaining a table of open ports, destination ids, and their corresponding ϵ values. The connection is generated when the base station calls the listen function and other nodes call connect. The base station broadcasts an INIT message to all network nodes to let them know of the new port listened to. The INIT message contains the port number and ϵ. The connect function on a port succeeds if the transport layer is already aware of this port via a past INIT. If the local node has not seen an INIT, a CONNECT message is sent out towards the destination with a given time-to-live (TTL), expressed in hops, to establish the connection. Forwarding of this message stops at the first node that is aware of the original INIT. An acknowledgment is sent back from that node to the connecting node indicating the point joining it to the connection tree.

After connection setup, the send function is used to reliably transmit general payload to the base station. This is done through hop-by-hop acknowledgments. In wireless reliable data transport, hop-by-hop loss recovery was shown to achieve much better performance in comparison with end-to-end recovery [1], [2]. Our transport protocol uses a message buffer for in-transit and received data packets. When receiving a packet from child nodes, the packet is acknowledged using a selective ACK message. The ACK message contains the port number and the sequence number of acknowledged packet. If the packet is destined to the receiver, it is added to the receive buffer to be fetched by the receive function. If not, it will be sent toward the base station.

B. Spatio-Temporal Data Transport

The specialized spatio-temporal data transport API employs the common send and receive mechanism. In addition, it has an estimation and error calculation mechanism to evaluate the importance of data. At every intermediate node v, the protocol maintains a local array of values representing $\bar{M}_v^v(t_{now})$. This array consists of t_{now} and values of $\bar{m}_w^v(t_{now})$ for every node w in T_v. For the sake of presentation we sometimes drop t_{now} from time-dependent variables to represent its current value. Later in this section, we shall describe how exactly $\bar{M}_v^v$ is maintained. The transport layer also keeps the last l successfully delivered estimated values to the parent, $\bar{M}_v^v(t_{s_1}), \ldots, \bar{M}_v^v(t_{s_l})$.

At every call to send_data function, the transport layer first updates $\bar{m}_v^v$ using the sample given by the application. Then, it creates a packet consisting the port number and the whole array representing $\bar{M}_v^v$. Based on the importance of that information, we derive a probability and using that, we send $\bar{M}_v^v$ to its parent. We call this probability $p_v(t)$. Note that in contrast to general packet transport, data is packed together at each hop and may be different from actual values sent from the sender.

We should use a value for p_v such that the final expected estimation error at the base station does not exceed $n\epsilon$. We use a local derivation and we show that this local decision will keep final error bounded regardless of global state of the network. The transport protocol first derives the estimated values in its parent if no new measurement is received by the parent. Formally, we compute $\bar{M}_v^u =$

$c_1\bar{M}_w^v(t_{s_1}) + \cdots + c_l\bar{M}_w^v(t_{s_l})$ using the physical model provided by the application. Then, the protocol computes F_v to be the squared difference between $\bar{M}_v^v$ and $\bar{M}_v^u$. F_v is the amount by which the estimation error at the parent node will be reduced if $\bar{M}_v^v$ is received by the parent. Using the expected deviation from last update, $E[F_v]$, we obtain p_v by:

$$p_v = 1 - \frac{\epsilon}{F_v} = 1 - \frac{\epsilon}{\sum_{w \in T_v} (\bar{m}_w^v - \bar{m}_w^u)^2} \quad (1)$$

If the wireless link between v and u is fully reliable, then the protocol just needs to send the packet one time with probability p_v. However, we need to accommodate faulty links in a way that the probability of receiving the message by u would be p_v. We discuss the details next in this section. Like traditional send functionality, an ACK message will be sent back to v if the packet is successfully received by u. Upon receipt of an ACK message, v knows that the estimated values in u has been updated to $\bar{M}_v^v$. The *send* function returns control to the application layer. The parent node does not directly forward the received packet to its parent. Rather, it uses the information to update values in $\bar{M}_u^u$. The packets would be collected in this fashion until the next call to send function at node u which probabilistically transmits the whole array of estimated data to the parent of u. The *receive* function at the basestation simply returns the $\bar{M}_r^r$.

In order to show that the expected estimation error at the basestation is indeed bounded by $n\epsilon$, we prove a stronger claim:

Theorem 1: The expected error that an intermediate node, v introduces to its parent, u, is less than or equal to $|T_v|\epsilon$.

Proof:

Our proof is by induction on height of v. Suppose v has the height of 0 which means that v is a leaf node and $T_v = \{v\}$(see Figure 1). Hence, the estimated data at v equals its own measurements, i.e. $M_v = \bar{M}_v^v = m_v$. The estimation error contributed to u is zero if the packet is received by u or it is $F_v = (\bar{m}_v^u - m_v)^2$ otherwise. Therefore, the expected value of error introduced to the parent equals $(1 - p_v)F_v$ which equals ϵ from (1).

Now, consider a node v with height $h + 1$. According to induction hypothesis, each child node, w_i, contributes $|T_{w_i}|\epsilon$ to the estimation error at v. This means that:

$$E_v = \sum_{w \in T_v} E[(\bar{m}_w^v - m_w)^2] = \sum_{1 \le i \le k} \sum_{j \in T_{w_i}} (\bar{m}_j^v - m_j)^2 \le \sum_{1 \le i \le k} |T_{w_i}|\epsilon = (|T_v| - 1)\epsilon \quad (2)$$

Using Bayes rule, we know that for every $w \in T_v - v$:

$$E[(\bar{m}_w^u - m_w)^2] = E[(\bar{m}_w^u - m_w)^2 | D_v = 1]P(D_v = 1) + E[(\bar{m}_w^u - m_w)^2 | D_v = 0]P(D_v = 0) \quad (3)$$

Algorithm 1 send_data(port_number, destination_id, payload)

1: Receive the application spatio-temporal data payload
2: $\bar{M}_v^u \leftarrow c_1\bar{M}_w^v(t_{s_1}) + \cdots + c_l\bar{M}_w^v(t_{s_l})$
3: $F_v \leftarrow \sum_{w \in T_v} (\bar{m}_w^u - \bar{m}_w^v)^2$
4: $p_v \leftarrow 1 - \frac{\epsilon}{E[F_v]}$
5: Compute q_v and r_v according to (8)
6: Transmit the data packet with probability q_v
7: Retransmit unless see the ACK from the parent node or reach r_v times
8: **return** control to application layer

where D_v is a random variable representing the delivery of packet from v to u. Given a successful delivery from v to u, $\bar{m}_w^u$ updates to $\bar{m}_w^v$. Therefore, $E[(\bar{m}_w^u - m_w)^2 | D_v = 1] = E[(\bar{m}_w^v - m_w)^2]$. On the other hand, depending on whether $\bar{m}_v^u$ has been updated to m_w or not we can write:

$$E[(\bar{m}_w^u - m_w)^2 | D_v = 0] = E[(\bar{m}_w^u - m_w)^2 | D_v = 0, U_v = 0]P(U_v = 0) + E[(\bar{m}_w^u - m_w)^2 | D_v = 0, U_v = 1]P(U_v = 1) \quad (4)$$

where U_v is one if $\bar{m}_v^u$ has been updated or zero otherwise. Again given $U_v = 1$, $\bar{m}_w^v = m_w$ and therefore $E[(\bar{m}_w^u - m_w)^2 | D_v = 0, U_v = 1] = E[\bar{m}_w^u - \bar{m}_w^v]$. Otherwise, the situation is that both $\bar{m}_w^u$ and $\bar{m}_w^v$ are not affected by new measurement at w meaning that $(\bar{m}_w^u - \bar{m}_w^v)^2$ and $(\bar{m}_w^v - m_w)^2$ are uncorrelated in this specific case. Thus, we can write $E[(\bar{m}_w^u - m_w)^2 | D_v = 0, U_v = 0] = E[(\bar{m}_w^v - m_w)^2] + E[(\bar{m}_w^u - \bar{m}_w^v)^2]$. From equations 3 and 4, we conclude that

$$E[(\bar{m}_w^u - m_w)^2] = E[(\bar{m}_m^v - m_w)^2]P(D_v = 1) + [E[(\bar{m}_w^v - m_w)^2]P(U_v = 0) + E[(\bar{m}_w^u - \bar{m}_w^v)^2]]P(D_v = 0) \le E[(\bar{m}_w^v - m_w)^2] + E[(\bar{m}_w^u - \bar{m}_w^v)^2]P(D_v = 0) \quad (5)$$

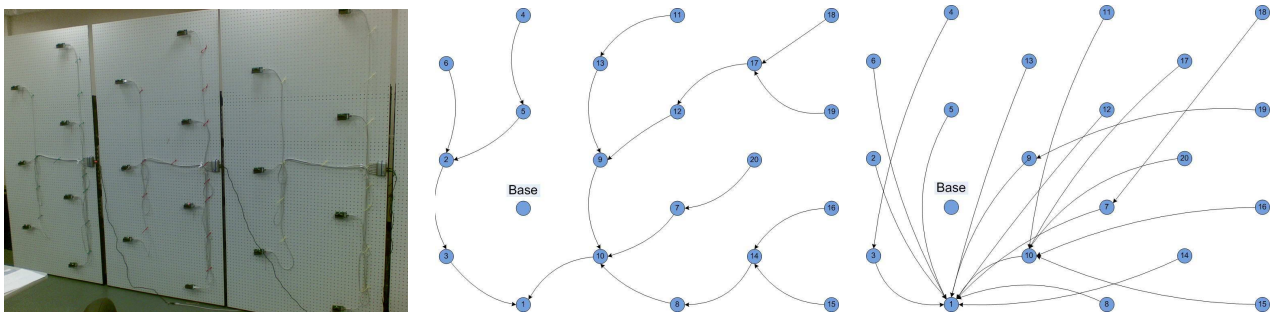
Now, summing both sides for all nodes in T_v and using (2) and the fact that $P(D_v = 0) = 1 - p_v = \frac{\epsilon}{F_v}$, we conclude:

$$\sum_{w \in T_v} E[(\bar{m}_w^u - m_w)^2] \le \sum_{w \in T_v} E[(\bar{m}_w^v - m_w)^2] + \sum_{w \in T_v} E[(\bar{m}_w^u - \bar{m}_w^v)^2]P(D_v = 0) \le (|T_v| - 1)\epsilon + F_v P(D_v = 0) = (|T_v| - 1)\epsilon + F_v\frac{\epsilon}{F_v} = |T_v|\epsilon \quad (6)$$

■

C. Channel Failures

As we discussed above, p_v is the probability at which the message from v should be delivered to u. When the link



(a) (b) (c)

Figure 2. The LiteOS testbed: (a) Physical configuration , (b) Routing tree with range $r = 1$, and (c) Routing tree with range $r = 2.646$.

is 100% reliable, a node just needs to send the packet with probability p_v. However, if the link is not reliable, we need to design a retransmission mechanism to make sure that the packet will be received by u with probability p_v. We denote the the delivery ratio of channel between u and v by d_v. We first assume that d_v is known to our protocol and later we explain how to estimate its value dynamically.

Depending on value of d_v, we can have two cases: First, if $p_v < d_v$ we derive $q_v = \frac{p_v}{d_v}$ and send the packet with probability q_v instead of p_v. This packet is delivered to u with probability of $q_v d_v = p_v$. In the second case, we have $p_v \geq d_v$ and therefore we need multiple transmissions to reach the requested p_v. If the protocol initiate transmission with probability q_v and perform r_v retransmissions to ensure delivery probability of p_v we can write:

$$p_v \leq q_v(1 - (1 - d_v)^{r_v}) \tag{7}$$

Now, we need to find the two values of q_v and r_v that satisfies the above equation. Due to limited computational capacity of current motes which are running this protocol and to avoid complex floating point operations we fix $q_v = 1$ and find the minimum value for r_v such that:

$$(1 - d_v)^{r_v} \leq 1 - p_v \tag{8}$$

In order to derive r_v from (8), the protocol multiplies $(1-d_v)$ by itself until the resulting value is less than $1-p_v$. This would ensure delivery probability of at least p_v. In summary, the transport protocol starts the transmission with probability q_v, and retries for r_v times until it receives an acknowledgment. Algotithm 1 summarizes different steps in the send procedure.

Until now, we assumed that our protocol knows the link delivery rate, d_v. In practice, we need to find d_v adaptively based on channel conditions. The idea is to use ACK messages for each transmission to update the value of d_v using an exponential moving average. In particular, let $a(k)$ be 1 if packet k has been acknowledged and 0 otherwise. We start with a small value for d_v and update it after transmitting every packet according to:

$$d_v(t) = (1 - \alpha)d_v(t - 1) + \alpha a(t) \tag{9}$$

IV. Experimental Testbed

We experimented with our protocol using a testbed of 21 MicaZ motes [14] presented in Figure 2(a). We used the LiteOS [15] operating system for development and implemented the protocol in C++. We use one of the motes as the base station. Each node in the testbed has a unique preset ID between 0 and 255. We used fixed tree topologies for our experiments where each topology is obtained by considering different communication ranges. Nodes in Figure 2(a) have roughly the same distance from their nearest neighbors. For simplicity of presentation, we choose the unit of distance to be the distance between two neighbor motes. Therefore, communication range of 1 is the minimum range to keep the network connected. We produce different topologies with communication ranges between 1 and 4.6. Two examples are illustrated in Figures 2(b), 2(c).

Our transport layer is implemented as a C++ class with *connect*, *listen*, *send*, *send_data*, *receive*, and *close* functions as described in Section III. In our implementation, transport layer buffer can hold up to 8 messages, 47 bytes each, considering extremely limited memory available in MicaZ motes. We implement an application, where each node independently and asynchronously reads the value of its light sensor at intervals of R seconds (i.e. sampling intervals) and uses our new protocol to send it. The observations are scalar values with the average of 700. We use $l = 1$ and $c_1 = 1$ for physical model.

In order to compare our protocol against other reliable and unreliable transport approaches, we implemented RMST [2] transport protocol based on the same framework as described above. The protocol uses the same two threads to send and receive packets. Each packet is retransmitted based on NACK received from the parent node or until we reach the maximum number of retries. Similarly, an unreliable protocol is implemented by simply sending each packet once to the parent; regardless of the delivery of the packet. We also implemented ESRT [6] by supplying the

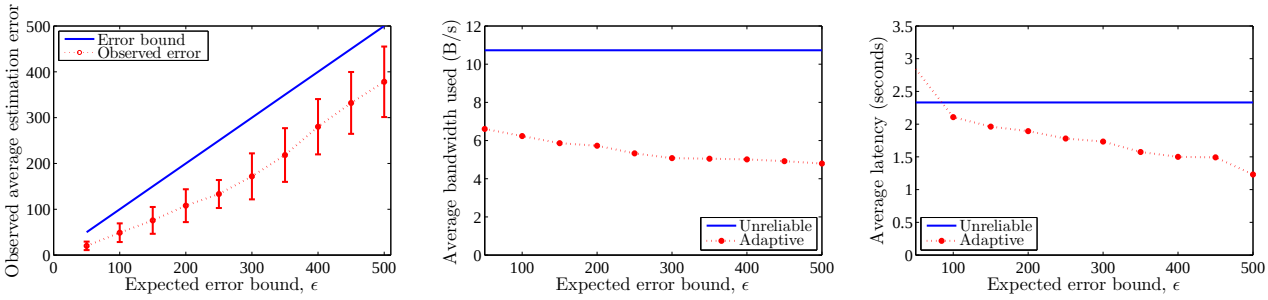


Figure 3. Impact of ϵ on protocol performance: (a) Average estimation error, (b) Communication overhead, and (c) Average delay.

unreliable framework with variable data reporting frequency. This frequency is adapted to maintain expected estimation error at base station. Unless otherwise specified, we use the following parameters for our experiments: The sampling interval, R is 5 seconds and ϵ equals 100. The average loss rate of links is 50%. We run the experiment for 50 cycles and report the average values.

V. Evaluation

In this section, we evaluate our protocol reliability, communication overhead and latency and compare those against different transport approaches. Although adaptive sampling and data reduction techniques use predictive physical models to provide bounded error, they are restricted to reliable networks and have not been implemented on extremely resource constrained motes. Therefore, no meaningful comparison with them is possible. We choose reliable transport protocols to compare against because they are designed to operate in unreliable medium where the delivery of packets is ultimately decided by the channel and not the protocol.

The average estimation error is computed by comparing the set of samples stored at the base station with the actual readings stored at each mote. An integrated mean square error is calculated and then divided by the number of motes and time to obtain average estimation error per node and unit time. The number of bytes each mote sends is also stored to calculate the communication overhead which is the total number of bytes sent divided by the number of motes and time. Since transmission is the only major energy consuming operation controlled by transport layer, we use average bandwidth used per node to represent both bandwidth and energy consumption of protocols. Use of any sleep/wake mechanism concurrently with the transport protocol can give our protocol more advantage on energy savings considering that data is not forwarded at arbitrary time instances but at predefined sampling intervals. To conduct a fair comparison, we do not use any sleep/wake mechanism in experiments.

The latency is computed by comparing the actual time at which a sample is taken and when the corresponding space-time sample is obtained at the base station. We compare these properties of the protocol against non-reliable and reliable protocols. First, we present a performance validation of our protocol to show that it can indeed provide the desired level of reliability. Next, we compare the protocols under various system parameters.

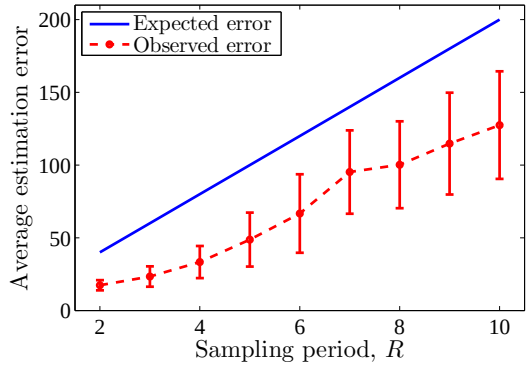


Figure 4. Impact of R on average estimation error.

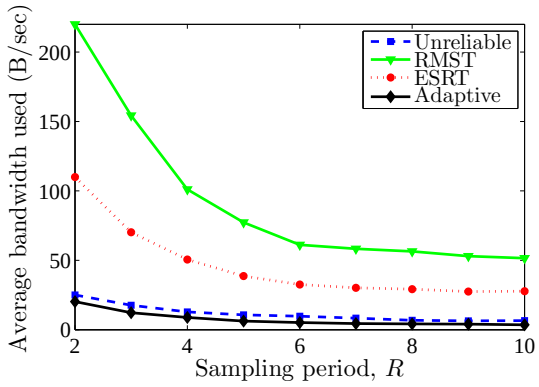
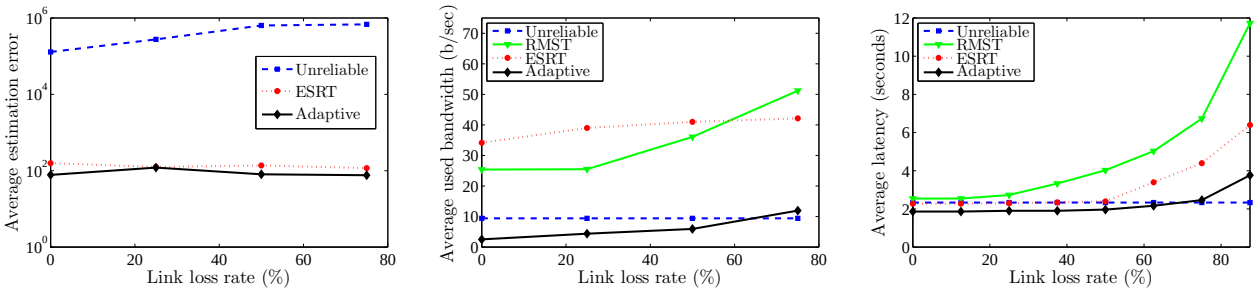


Figure 5. Comparing estimation error for various sampling rates.

We validate performance of our protocol against the two protocol parameters ϵ and R. In the first experiment, we study the impact of ϵ on the average estimation error



(a) (b) (c)

Figure 6. Comparing (a) estimation error, (b) communication overhead, and (c) average latency for various channel qualities.

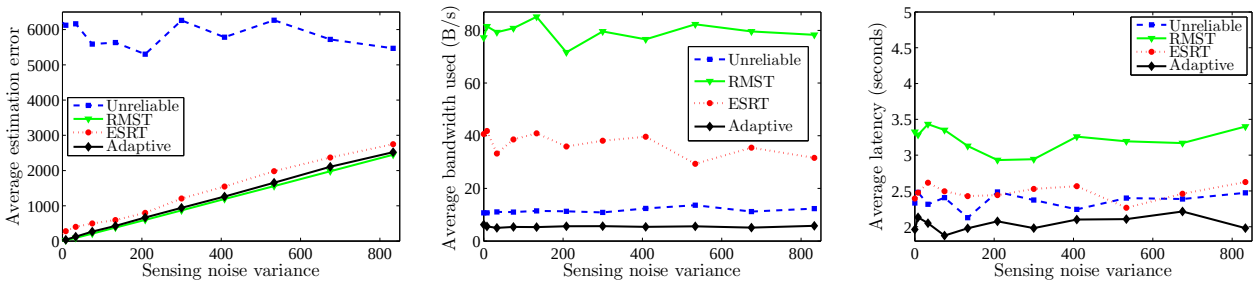
observed by the base station. We change ϵ between 50 and 500 and report the average and standard deviation of estimation error. We plot the result in Figure 3(a). Our protocol guarantees that the average estimation error is always less than or equal to ϵ. Therefore, we expect the error curve to be below the $y = x$ line. On the other hand, our goal is to have the curve as close as possible to $y = x$, to save as much energy as possible. The graph shows that our protocol is rather close the ideal case. Next, we collect the number of bytes transmitted and their latencies in the same experiment and show it in Figures 3(b) and 3(c). We plot the communication overhead of unreliable data transport as a reference. As we increase ϵ, both communication overhead and latency drop as fewer transmissions are required to maintain the error level.

In the next experiment, we change the cycle length of the protocol, R, between 2 and 10 seconds and report the average estimation error at sample points in Figure V. Increasing R increases the time interval between samples and therefore increases integrated mean square of error over time and the average estimation error. However, this implies less samples to be transmitted and therefore less communication overhead. We plot the average number of bytes transmitted per node in Figure V and see that the communication overhead increases super linearly with decreasing R. The reason is that decreasing R means transmitting more packets in a time unit and therefore increased chance of interference or collisions.

In the first comparison experiment, we study the performance of our protocol in various channel conditions compared to RMST, ESRT, and non-reliable transport protocols. We change the link delivery rate between 25% to 100% for all links and present the results in Figure 6(b). The number of packets our protocol sends is less than 20% of what RMST and ESRT would send. This means savings of about 80% in energy consumption. Note that, the number of packets transmitted by a non-reliable protocol does not depend on link quality since it always sends one packet per reading. Next, we observe the estimation error at the base station while we change the link delivery rate (Figure 6(a)). Since RMST always provides an estimation error of zero and the y-axis is plotted in log scale, we omit the RMST curve. The plot shows that the estimation error provided by our protocol is at least three orders of magnitude less than an unreliable approach. Finally, the latency comparison is presented in Figure 6(c). By increasing link loss ratio, a huge latency is contributed to RMST because of the increasing number of retransmissions and collisions.

In the next experiment, we compare the protocols as we change the noise on sensor readings. To model white noise, each node generates a normal random value with mean 0 and variance σ^2 at the time of sampling and adds it to the actual light reading. We first run the experiment for different values of σ^2 and report the average estimation error at the base station. The error is computed by subtracting the base station values and real sensed values at nodes for each time instance and plotted in Figure 7(a). Since adaptive and reliable transport protocols provide error-free or low-error communication, the final estimation error is approximately the same as noise. However, the unreliable transport protocol communication error dominates the noise and shows an almost constant error. We perform the same experiment to study the communication overhead and latency. We record the number of packets sent by each node and find the average bandwidth used. Figures 7(b) and 7(c) show that the sensing noise does not have any effect on the communication overhead and latency as sensing noise is completely outside the scope of the transport protocols.

In the last set of experiments, we compare transport protocols for various communication ranges and tree topologies. We fix the position of the motes and change the range they can directly communicate. In one extreme, any node can only communicate with its nearest neighbor, while on the other extreme, the communication range is large enough to make every node able to communicate directly with the base station. For different values of communication range we use a fixed corresponding routing tree and ran the experiment for the three different protocols. We first plot the estimation error incurred at the base station in Figure 8(a). Note that, the y-axis is shown in log scale for clarity; therefore, we



(a) (b) (c)

Figure 7. Comparing transport protocols with different noises: (a) Average estimation error, (b) Communication overhead, and (c) Average delay.

don't plot RMST performance which constantly zero. The results show that our transport protocol keeps the same error level as we aim to guarantee.

Next, we plot the communication overhead of the protocols in Figure 8(b). Increasing transmission range increases the communication overhead in adaptive and reliable transport protocols. The reason is that increasing transmission range would increase the interference and degrade the link quality which eventually increases the number of retransmissions. On the other hand, fewer hops do not decrease the communication overhead since we use a constant packet size and aggregation in intermediate nodes. Finally, we illustrate the average latency as we change the communication range in Figure 8(c). Increasing communication range results in shorter paths and therefore, the average latency decreases for all protocols. All results show that the adaptive transport protocol can provide a guaranteed error rate using much less resources than any other protocol.

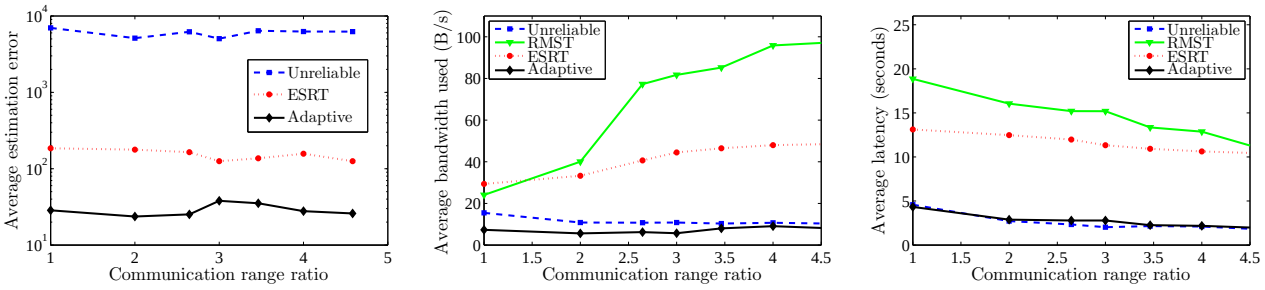
VI. Related Work

There have been several studies on point-to-point reliable transport protocols for multi hop wireless networks [3]–[5], [16], [17]. The authors of [16] propose a receiver centric transport protocol (RCP) where the receiver controls the transmission rate and achieves high throughput by accommodating the lossy nature of wireless links and use of heterogeneous interfaces. In [17], authors redesign TCP and its variants for wireless ad hoc networks by changing window-based transmission and decoupling congestion indication and packet loss. A new receiver-initiated reliable transport protocol (Flush) is proposed in [3] which guarantees reliability using selective NACKs and dynamically controls transmission rate by estimating interference.

Authors in [18] study the packet-level reliability semantics through end-to-end retransmission, error correction codes, link-level retransmissions, and route fixing to show that a correct combination of these mechanisms can provide very high packet-level reliability. Deluge protocol [19] is proposed to reliably transfer large data objects in a wireless sensor network while adapting to spatial node density. [20] presents a reliable sink-to-sensor data transport protocol. The paper defines various one-to-many reliability requirements based on sub-network coverage areas. For example, some message needs to be received by all nodes in the network while another type of message requires only nodes in an specific region.

Several other works discussed many-to-one transport protocols [1], [2], [5]. In [2], authors propose RMST, a reliable multi-segment transport which is designed to run on top of directed diffusion and uses hop-by-hop NACKs to guarantee reliability. In [1], the PSFQ protocol has been proposed with focus on how to provide end-to-end reliability efficiently using hop-by-hop NACK messaging. More closely to our work, some works address many-to-one communication paradigms with new reliability semantics specific to wireless sensor networks [6], [7]. Authors in [6] propose a protocol to report an event observed by sensors to the base station (ESRT). This protocol assumes that the base station needs a minimum number of reports from sensors to reliably identify the event and therefore it adapts transmission rate of sensors using a feedback loop. Similarly in [7], authors propose PORT and define reliability based a function of the packet rate received from each node. Given a threshold on packet rate, the protocol minimizes the number of messages being transferred through a feedback mechanism. Authors in [8], propose a real-time and reliable transport protocol focusing on several characteristics of different applications such as number of packets being received from the event and the delay constraint of decision interval. All of proposed protocols have a very limited knowledge about the real importance of a data packet from the application perspective. Unlike those, we introduce a protocol designed efficiently for spatio-temporal data collection. Our approach provides accurate estimation with much lower communication overhead.

The protocols for many-to-one data transport focusing on congestion control problem rather than reliability semantics have been studied in [3]–[5], [21]. The authors of [21] address the congestion control problem in a many-to-one routing scheme by performing rate control at each node in



(a) (b) (c)

Figure 8. Comparing different protocols for various communication ranges: (a) Average estimation error, (b) Communication overhead, and (c) Average delay.

a distributed manner. Each node notifies its predecessors in a converge-cast tree about its maximum transfer rate and adjust its own based on what received from the successor. In [5], authors study the reverse problem, namely congestion control problem in the downstream direction from sink to sensor. This paper provides a rate control mechanism to reduce workload.

Data aggregation is one of the main characteristics of wireless sensor networks and has been addressed in several works [13], [22]. [22] presents Directed Diffusion, a data-centric dissemination approach at which data can be directed toward the sink on several different paths and intermediate nodes can merge two or more samples that have been received and report a single aggregate value. Authors in [13] present TAG, an aggregation service for sensor networks using a query system. Each query asks for a periodic report of an aggregate value in the network. The base station broadcasts the query and leaf nodes generate requested data and transmit it to the sink through an aggregation tree while intermediate nodes compute the aggregate value using the appropriate function. Comparing to our work, these does not address the issue of reliability. In [23], authors formulate the many-to-one data transport problem given specific channel constraint and message utility functions. Using this formulation, they provide a framework for studying the optimal flow assignment to maximize the total utility of the network.

Estimation theory and predictive modeling has been used widely in monitoring applications to reduce the amount of data needs to be transferred in a sensor network [9]–[11]. In [9], authors propose a mechanism to decide when new data packets should be sent based on the local variation in their readings. Using this scheme, only nodes with high difference in recent observations would transmit their readings and therefore, energy is saved. In [10], another energy saving method is proposed for field estimation in which sampling is performed with a lower resolution in the areas with less variability of the physical phenomenon.

Authors in [24], discuss the problem of selecting a small subset of sensors to achieve the lowest estimation error given the energy constraint on the network. [11] proposes a data reduction protocol using an adaptive physical model to save on transmitting the predictable data. A learning algorithm is employed at each node to fit the linear model to collected samples. Then, a sample is dropped at the sender if it can be accurately estimated from the learned model. Similarly, [12] uses autoregressive models to predict sensor readings given a query at the sink. In comparison to our scheme, all these works are aimed toward reducing communication overhead in a reliable medium. They can not work in an unreliable environment where our protocol is designed to provide bounded error through probabilistic retransmissions. Furthermore, to the best of our knowledge, there is no data reduction technique implemented on motes.

VII. Conclusion

We proposed new reliability semantics for spatio-temporal data transport in wireless sensor networks performing monitoring applications. We used estimation error as a metric to determine the amount of information that is to be successfully delivered. Based on this metric, we proposed a light-weight transport protocol to achieve the desired level of estimation error with a very low communication overhead and energy consumption. The protocol monitors wireless channel quality in order to adapt to variable link conditions. We implemented our protocol in LiteOS on MicaZ motes and rigorously evaluated its performance through experiments. Our protocol achieved the desired reliability with minimal latency and energy consumption. The protocol does not require feedback from the base station or synchronization among nodes. We compare our protocol against RMST, ESRT, and unreliable transport protocols and show that we save a great amount of energy at the expense of producing a small error in the output.

References

[1] C.-Y. Wan, A. T. Campbell, and L. Krishnamurthy, "PSFQ: a reliable transport protocol for wireless sensor networks," in *Proc. of the 1st ACM International Workshop on Wireless Sensor Networks and Applications (WSNA'02)*. New York: ACM, 2002, pp. 1–11.

[2] F. Stann and J. Heidemann, "RMST: reliable data transport in sensor networks," in *Proc. of the 1st IEEE International Workshop on Sensor Network Protocols and Applications (SNPA'03)*, 2003, pp. 102–112.

[3] S. Kim, R. Fonseca, P. Dutta, A. Tavakoli, D. Culler, P. Levis, S. Shenker, and I. Stoica, "Flush: A reliable bulk transport protocol for multihop wireless networks," in *Proc. of the 5th ACM Conference on Embedded Networked Sensor Systems (SenSys'07)*. New York: ACM, 2007, pp. 351–365.

[4] J. Paek and R. Govindan, "RCRT: rate-controlled reliable transport for wireless sensor networks," in *Proc. of the 5th ACM Conference on Embedded Networked Sensor Systems (SenSys'07)*. New York: ACM, 2007, pp. 305–319.

[5] R. Vedantham, R. Sivakumar, and S.-J. Park, "Sink-to-sensors congestion control," in *Proc. of IEEE International Conference on Communications (ICC'05)*, vol. 5, May 2005, pp. 3211– 3217.

[6] Y. Sankarasubramaniam, Özgür B. Akan, and I. F. Akyildiz, "ESRT: event-to-sink reliable transport in wireless sensor networks," in *Proc. of the 4th International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'03)*, 2003, pp. 177–188.

[7] Y. Zhou, M. Lyu, J. Liu, and H. Wang, "PORT: a price-oriented reliable transport protocol for wireless sensor networks," in *Proc. of the 16th IEEE International Symposium on Software Reliability Engineering*, 2005, pp. 10–.

[8] V. C. Gungor, Özgür B. Akan, and I. F. Akyildiz, "A real-time and reliable transport (rt) 2 protocol for wireless sensor and actor networks," *IEEE/ACM Transactions on Networking*, vol. 16, no. 2, pp. 359–370, 2008.

[9] M. Chu, H. Haussecker, and F. Zhao, "Scalable information-driven sensor querying and routing for ad hoc heterogeneous sensor networks," *International Journal of High Performance Computing Applications*, vol. 16, no. 3, pp. 203–313, 2002.

[10] R. Nowak, U. Mitra, and R. Willett, "Estimating inhomogeneous fields using wireless sensor networks," *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 6, pp. 999–1006, 2004.

[11] S. Santini and K. Roemer, "An adaptive strategy for quality-based data reduction in wireless sensor networks," in *Proc. of the 3rd International Conference on Networked Sensing Systems (INSS06)*, 2006, pp. 29–36.

[12] D. Tulone and S. Madden, "PAQ: time series forecasting for approximate query answering in sensor networks," in *Proc. of the 3rd European Workshop on Wireless Sensor Networks (EWSN'06)*, 2006, pp. 21–37.

[13] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "TAG: a tiny aggregation service for ad-hoc sensor networks," *SIGOPS Operating Systems Review*, vol. 36, no. SI, pp. 131–146, 2002.

[14] MicaZ Data Sheet, "http://xbow.com/Products/Product_pdf_files/Wireless_pdf/MICAz_Datasheet.pdf."

[15] LiteOS homepage, "http://www.liteos.net/."

[16] K.-H. Kim, Y. Zhu, R. Sivakumar, and H.-Y. Hsieh, "A receiver-centric transport protocol for mobile hosts with heterogeneous wireless interfaces," *Wireless Networking*, vol. 11, no. 4, pp. 363–382, 2005.

[17] K. Sundaresan, V. Anantharaman, H.-Y. Hsieh, and R. Sivakumar, "ATP: A reliable transport protocol for ad hoc networks," *IEEE Transactions on Mobile Computing*, vol. 4, no. 6, pp. 588–603, 2005.

[18] S. Kim, R. Fonseca, and D. Culler, "Reliable transfer on wireless sensor networks," in *Proc. of the 1st IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks (SECON'04)*, 2004, pp. 449–459.

[19] J. W. Hui and D. Culler, "The dynamic behavior of a data dissemination protocol for network programming at scale," in *Proc. of the 2nd international conference on Embedded networked sensor systems (SenSys'04)*. New York, NY, USA: ACM, 2004, pp. 81–94.

[20] S.-J. Park, R. Vedantham, R. Sivakumar, and I. F. Akyildiz, "A scalable approach for reliable downstream data delivery in wireless sensor networks," in *Proc. of the 5th ACM international symposium on Mobile ad hoc networking and computing (MobiHoc'04)*. New York, NY, USA: ACM, 2004, pp. 78–89.

[21] C. T. Ee and R. Bajcsy, "Congestion control and fairness for many-to-one routing in sensor networks," in *Proc. of the 2nd ACM Conference on Embedded Networked Sensor Systems (SenSys'04)*. New York: ACM, 2004, pp. 148–161.

[22] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion: a scalable and robust communication paradigm for sensor networks," in *Proc. of the 6th International Conference on Mobile Computing and Networking (MobiCom'00)*. New York: ACM, 2000, pp. 56–67.

[23] W.-P. Chen and L. Sha, "An energy-aware data-centric generic utility based approach in wireless sensor networks," in *Proc. of the 3rd international symposium on Information processing in sensor networks (IPSN'04)*. Berkeley, CA: ACM, 2004, pp. 215–224.

[24] X. Huang and I. Rubin, "Capacity and energy aware activation of sensor nodes for area phenomenon using wireless network transport," in *Proc. of the 2007 international conference on Wireless communications and mobile computing (IWCMC'07)*, 2007, pp. 463–468.